Combined Quay Crane Assignment and Quay Crane Scheduling with Crane Inter-Vessel Movement and Non-Interference Constraints

Ghazwan Alsoufi, Xinan Yang, Abdellah Salhi
Department of Mathematical Sciences, University of Essex, Colchester, UK

Abstract

Integrated models of the Quay Crane Assignment Problem (QCAP) and the Quay Crane Scheduling Problem (QCSP) exist. However, they have shortcomings in that some do not allow movement of quay cranes between vessels, others do not take into account precedence relationships between tasks, and yet others do not avoid interference between quay cranes. Here, an integrated and comprehensive optimisation model that combines the two distinct QCAP and QCSP problems which deals with the issues raised is put forward. The model is of the mixed-integer programming type with the objective being to minimize the difference between tardiness cost and earliness income based on finishing time and requested departure time for a vessel. Because of the extent of the model and the potential for even small problems to lead to large instances, exact methods can be prohibitive in computational time. For this reason an adapted Genetic Algorithm (GA) is implemented to cope with this computational burden. Experimental results obtained with Branch-and-Cut (B&C) as implemented in CPLEX and GA for small to large scale problem instances are presented. The paper also includes a review of the relevant literature.

Keyword

Container Terminals, Quay Crane Assignment, Quay Crane Scheduling, Branch-and-Cut, Genetic Algorithm

1. Introduction

Container terminals are essential for the distribution of import/export goods to domestic markets around the world. Quay cranes are important and valuable equipment used in these terminals. They, therefore, must be used optimally whenever possible. Determining the optimum number of quay cranes to assign to a given vessel moored in a container port, and the optimum sequence in which to perform the tasks of unloading and loading the vessel constitute some of the most important operations at ports. These operations potentially lead to distinct problems as follows.

1. The Berth Allocation Problem (BAP);
2. The Quay Crane Assignment Problem (QCAP);
3. The Quay Crane Scheduling Problem (QCSP).

Comprehensive reviews of these three problems (seaside problems) individually and when integrated pairwise as in BAP&QCAP, and QCAP&QCSP, can be found in (Steenken et al., 2004; Bierwirth and Meisel, 2010). Each one of these problems considered individually, is complex enough. However, solving them individually and without consideration of the others may lead to suboptimal solutions. For this reason the pairwise integrated problems are investigated to obtain near optimal solutions.

After berth allocation, (Alsoufi et al., 2016), quay cranes must be assigned to each docked vessel. The number of quay cranes assigned to a vessel impacts on its processing time and therefore the working efficiency of the port. If the number of quay cranes assigned to a vessel is below requirement, the vessel will take unnecessarily longer processing time. This may result in a delay of the vessel's departure, which

will, potentially, have a knock-on effect on the allocation of berths to waiting vessels, and other port operations. Equally, if the number of assigned quay cranes is more than what is needed, it may lead to high handling cost for this vessel and potentially delay the processing of other vessels. The other problem faced by the decision maker after assigning quay cranes to vessels is finding the optimum sequence in which to perform tasks (unload and load containers from and onto vessels). This is the problem of scheduling quay cranes to carry out all tasks efficiently and without interference between them.

(Alsoufi et al., 2015) have highlighted the possibility of finding efficient working plans when all seaside operations (BAP, QCAP and QCSP) are considered together in a single model. However, the attempt is ambitious and important issues have not been taken into account. Experiments, for instance, were only on very small instances which do not match anything that is found in the real world even in small provincial container ports. These instances were solved with CPLEX only. Note that the instances of the model blow up very quickly with even the smallest increase in the number of vessels and/or quay cranes. The current work focuses on the combination of QCAP and QCSP into a model the solution of which does not require changing or adapting the berthing plan. The model is comprehensive, taking account into a number of issues such as allowing the movement of quay cranes between holds of the same vessel and holds on different vessels. The solution of instances of this model have been obtained with both B&C as implemented in CPLEX, but also an innovative implementation of the Genetic Algorithm.

QCAP and QCSP are closely related as the former feeds the latter with the number of quay cranes as input. Without the solution of the latter one cannot see the accurate processing time of a vessel given the fixed number of quay cranes assigned to it.

Few attempts at combining QCAP and QCSP, as in (Diabat and Theodorou, 2014; Fu et al., 2014; Theodorou and Diabat, 2015) have been made. This could possible by due to the complexity of the mathematical formulation and especially its computational demands. It is also the case that, to the best of our knowledge, previous works that combine QCAP and QCSP did not allow quay cranes to move between vessels while the latter are still being processed, (Daganzo, 1989; Peterkofsky and Daganzo, 1990; Tavakkoli-Moghaddam et al., 2009), for instance.

This paper focuses on the integration of QCAP and QCSP to form QCASP, an extended decision making problem that can be solved in real applications. Contributions are two fold:

1. to formulate a mathematical model that combines QCAP and QCSP and allows quay cranes to move between vessels while they are still being processed. In other words, it allows the number of quay cranes allocated to any vessel to change during the handling of the vessel;

2. to solve realistic instances of the problem using an adapted variant of the Genetic Algorithm (GA).

The paper is organised as follows. Section 2 is a literature review of QCSP, BAP combined with QCAP, and QCAP combined with QCSP. Section 3 presents a mathematical model that combines QCAP and QCSP. In Section 4, a variant of GA is suggested for QCASP. In Section 5, the computational results are reported and discussed. Section 6 concludes the paper and suggests further directions of research on this topic.

2. Container port operations: a review

As said earlier, assigning and scheduling quay cranes is at the heart of container port operations. Here, we review the relevant literature on these problems and related ones.

2.1. The Quay Crane Scheduling Problem (QCSP)

The solution of QCSP is the optimal sequence of moves for quay cranes to perform (unloading and loading containers from and unto the vessel) in order to minimize the handling time of the vessel.

(Kim and Park, 2004) studied QCSP and formulated it as a mixed integer programme. They used Branch-and-Bound (B&B) in conjunction with the Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende, 1995), to overcome the difficulties of B&B on its own. (Moccia et al.,

2006), formulated QCSP as a vehicle routing problem with additional constraints like the precedence relationships between tasks. CPLEX was used to solve small scale instances and the Branch-and-Cut (B&C) algorithm to solve large scale instances. (Sammarra et al., 2007) proposed a tabu search heuristic for QCSP in order to minimize the completion time for unloading and/or loading containers from/into vessels. They considered precedence and non-simultaneity between tasks. They also observed that QCSP can be decomposed into a routing problem and a scheduling problem. (Lee et al., 2008) presented a mixed integer programming model of QCSP and proved that it is NP-complete; they used a GA to solve problem instances to near optimality. (Bierwirth and Meisel, 2009) noticed the shortcomings of earlier models of QCSP, and in particular with respect to the quay crane interference avoiding constraints which did not do the job properly. They revised one such model due to (Sammarra et al., 2007) to take care of interference between quay cranes. They also proposed a Unidirectional Schedule (UDS) heuristic for when the quay cranes do not change moving direction from their initial position and have identical directions of movement both from upper and lower bays. (Chung and Choy, 2012) proposed a variant of GA to solve (Kim and Park, 2004) model's of QCSP. Their results compared well with those obtained by most well established algorithms. (Kaveshgar et al., 2012) introduced an efficient GA for QCSP. Their algorithm improved the efficiency of GA search by using an initial solution based on the S-LOAD rule and by reducing the number of genes in the chromosomes to reduce search time. (Nguyen et al., 2013) suggested two representations of QCSP one for GA and the other for Genetic Programming (GP), (Koza, 1992). GA uses permutation to decide the priority of tasks, whereas GP relied on a priority function to calculate the priority of tasks. (Unsal and Oguz, 2013) proposed a constraint programming approach for solving QCSP with container groups. (Kenan and Diabat, 2015) recast QCSP into a Dantzig-Wolfe formulation that can be solved by column generation. They then developed a Branch-and-Price procedure, which is an exact method, to effectively solve large instances of mixed integer programs.

2.2. The Combined Berth Allocation and Quay Crane Assignment Problem (BACAP)

QCAP is the problem of finding the optimum number of quay cranes that should be assigned to every vessel that docks at a container terminal. This problem can be seen as trivial since knowing the workload of a vessel and the work rate of a quay crane should allow to estimate the number of quay cranes required for the vessel. However, if quay cranes of different work rates are used and are allowed to move from one ship to another while work on the ships is ongoing, then their assignment is no longer so simple. BACAP is the problem of allocating berthing times and berthing positions to vessels and, at the same time, determining the optimum number of quay cranes to service them.

(Legato et al., 2008), addressed QCAP with a predetermined berth position and time following the solution of BAP. They assumed that quay cranes could not move between vessels before all tasks are performed and the vessels processing is completely finished. A mathematical model was presented to determine the optimum number of quay cranes for each vessel that is ready for processing. (Meisel and Bierwirth, 2009) integrated BAP, and QCAP into BACAP. The proposed problem is formulated taking into account some of the real issues faced by the decision maker at the port. In addition to the mathematical model, they also suggested two meta-heuristic approaches for the problem: the Squeaky Wheel Optimization (SWO), and Tabu Search (TS). (Cheong et al., 2010) considered the multi-objective optimization aspect of BACAP; indeed, it involves simultaneous optimization of two highly-coupled container terminal operations. Optimization results show that the multi-objective approach offers the port manager flexibility in selecting a desirable solution for implementation. (Yang et al., 2012) suggested to deal with BACAP by solving simultaneously BAP and QCAP. They formulated a mathematical model which integrates the BAP constraints of (Guan and Cheung, 2004) and the QCAP constraints of (Legato et al., 2008). The objective function for this model is the combination of the objective functions of the BAP and QCAP models. An evolutionary algorithm was developed to find the solution to this coarse combined problem.

2.3. The Combined Quay Crane Assignment and Quay Crane Scheduling Problem (QCASP)

The decision maker in QCASP focuses on determining the number of quay cranes for each vessel and finding the best sequence in which tasks will be performed by these quay cranes. There is obviously a strong relationship between the two goals.

(Daganzo, 1989) addressed quay crane scheduling for multiple vessel arrivals. They proposed both an exact and an approximate solution approach with the objective being to minimise the tardiness of all vessels. (Peterkofsky and Daganzo, 1990) developed a B&B algorithm to solve QCSP. Both of these papers did not consider the interference between quay cranes. (Tavakkoli-Moghaddam et al., 2009) studied QCASP. They formulated a mixed integer programme to determine the optimal number of quay cranes for every vessel that will arrive at the terminal and at the same time the optimal sequence in which the tasks should be carried out on the vessel. An evolutionary approach (GA) is suggested to solve large scale instances of this type of problem. (Unsal and Oguz, 2013) extended their constraint programming model for QCSP and converted it into an integrated Quay Crane Assignment and Scheduling problem (QCASP) in which QCs are assigned to vessels and scheduled to work on multiple vessels, based on a berthing plan of vessels. (Diabat and Theodorou, 2014; Fu et al., 2014) also proposed a combined model for QCASP and implemented a variant of GA to solve large scale instances of the problem. Their model allows the movement of quay cranes between vessels while the processing of vessels is ongoing. This is achieved by discretizing the time horizon and using a time index. The interference avoiding constraint is represented simply by the position of quay cranes at each short time interval. This increases the number of variables in the model and potentially makes the problem difficult to solve (curse of dimensionality). Note that time discretization makes the model less accurate. The omission of quay crane travelling time between bays/vessels makes the solution of (Fu et al., 2014) impractical, especially when the frequency of quay crane movement is high. Precedence and simultaneity constraints are also not considered. Note that the results recorded in Table 3 of (Fu et al., 2014), for instance, do not make sense. They used GAMS, the General Algebraic Modeling System, a commercial software for solving optimization problems. However, the results returned by GAMS which are presumably exact, cannot possibly be worse than those returned by GA for most of the problem instances considered. (Theodorou and Diabat, 2015) presented an approach to solve QCASP by transforming it first into a crane-to-bay assignment problem, and then applying a Lagrangian relaxation algorithm to it.

3. Mathematical formulation

This section describes a mixed integer programming model of QCASP, the solution of which is the optimum number of quay cranes to be assigned to each docked vessel and their scheduling to carry out all necessary moves with the objective being to minimise processing costs. Note that the number of quay cranes assigned to a vessel at the beginning of the operation may not be the same as at the end because our model allows quay cranes to move between vessels. Comparing with handling these two problems individually, the combined model as proposed here, does not require us to estimate the processing time when allocating quay cranes to vessels. Therefore, it allows in general to find more accurate solutions.

The improvement on solving QCAP and QCSP in succession, is that we do not force the number of quay cranes allocated to a vessel to be fixed during the whole processing period of the vessel. If necessary, a quay crane may move from one vessel to another before the processing of this vessel has finished. This is more flexible than using a fixed number of quay cranes to handle a vessel; it has the potential to give better working plans (Alsoufi et al., 2015), and also, to some extent, to mitigate uncertainty linked to the performance of quay cranes.

3.1. Assumptions

Consider a continuous berth container terminal with fixed length and berth allocation already decided. Now assume that:

1- the berthing position and berthing time of vessels are given as inputs;
2- each vessel is divided longitudinally into bays; all bays have the same length. Thus, the length of a vessel is given as the number of bays;
3- the safety distance between each pair of adjacent quay cranes depends on the width of a bay;
4- once a quay crane starts processing a task, it can leave only after it has finished the workload of this task;
5- quay cranes are on the same rail and thus they cannot cross over each other;
6- some tasks must be performed before others and there are some tasks which cannot be performed simultaneously.

3.2. Indices

Q	Number of quay cranes $(q, q_i, q_j = 1, 2, ...Q)$.
V	Number of vessels $(v, v_i, v_j = 1, 2, ..., V)$.
B_v	Number of tasks on vessel v $(b, b_i, b_j = 1, 2, ..., B_v)$.

3.3. Parameters

p_b^v	Time required to perform task b on vessel v.
l_b^v	Location of task b on vessel v expressed by the ship bay number on vessel v.
r_q	Earliest available time of the q^{th} quay crane.
I_0^q	Initial location of quay crane q which is relative to the ship-bay number.
$t_{b_i b_j}^{qv}$	Travel time of the q^{th} quay crane from the location $l_{b_i}^v$ of task b_i to the location $l_{b_j}^v$ of task b_j. $t_{b_0 b_j}^{qv}$ Represents the travel time from the initial position I_0^q of the q^{th} quay crane to the location $l_{b_j}^v$ of the task b_j on vessel v.
T_v	Berthing time of vessel v.
d_v	Requested departure time for vessel v.
W_v	Tardiness cost (per-hour) of vessel v per time unit.
R_v	Earliness income of vessel v per time unit .
Ψ	Set of pairs of tasks that cannot be performed simultaneously. When tasks b_i and b_j cannot be performed simultaneously, then $(b_i, b_j) \in \Psi$.
Φ	Set of ordered pairs of tasks for which there is a precedence relationship. When task b_i must precede task b_j, then $(b_i, b_j) \in \Phi$.
M	Arbitrary large positive number.

3.4. Binary decision variables

$$X_{b_i b_j}^{qv} = \begin{cases} 1 & \text{if the } q^{th} \text{ quay crane performs task } b_j \text{ immediately after performing task } b_i \text{ on vessel } v. \\ 0 & \text{otherwise} \end{cases}$$

Tasks b_0 and b_{B_v+1} are considered as the dummy initial and final states of each quay crane, respectively. Thus, when task b_j is the first task of the q^{th} quay crane on vessel v then $X_{b_0 b_j}^{qv} = 1$. Similarly, when task b_j is the last task of the q^{th} quay crane on vessel v then $X_{b_j b_{B_v+1}}^{qv} = 1$.

$$Z^{v}_{b_i b_j} = \begin{cases} 1 & \text{if task } b_j \text{ starts later than the finish time of task } b_i \text{ on vessel } v. \\ 0 & \text{otherwise.} \end{cases}$$

$$Y^{q}_{v_i v_j} = \begin{cases} 1 & \text{if the } q^{th} \text{ quay crane is assigned to vessel } v_j \text{ immediately after finishing its task on vessel } v_i. \\ 0 & \text{otherwise.} \end{cases}$$

$$\alpha^{v_i v_j}_{b_i b_j} = \begin{cases} 1 & \text{if task } b_j \text{ on vessel } v_j \text{ is located below task } b_i \text{ on vessel } v_i. \\ 0 & \text{otherwise.} \end{cases}$$

$$\beta^{v_i v_j}_{b_i b_j} = \begin{cases} 1 & \text{if task } b_j \text{ on vessel } v_j \text{ starts later than the finish time of task } b_i \text{ on vessel } v_i \\ 0 & \text{otherwise.} \end{cases}$$

3.5. Continuous decision variables

E_v	Earliness of vessel v.
A_v	Tardiness of vessel v.
S_{qv}	Starting time of q^{th} quay crane on vessel v.
$D^{v}_{b_i}$	Completion time of task b_i on vessel v.
C_{qv}	Completion time of q^{th} quay crane on vessel v.
F_v	Finishing (departure) time of the vessel v.

3.6. The mathematical model

$$\min(Z) = \sum_{v=1}^{V} W_v A_v - \sum_{v=1}^{V} R_v E_v \tag{1}$$

s.t

$$d_v - F_v = E_v - A_v, \qquad \forall v \tag{2}$$

$$\sum_{v_j=1}^{V} Y^{q}_{v_0 v_j} = 1, \qquad \forall q \tag{3}$$

$$\sum_{v_i=1}^{V} Y^{q}_{v_i (V+1)} = 1, \qquad \forall q \tag{4}$$

$$\sum_{v_j=1}^{V+1} Y^{q}_{v v_j} - \sum_{v_j=0}^{V} Y^{q}_{v_j v} = 0, \qquad \forall v, q \tag{5}$$

$$\sum_{v_i=0}^{V} \sum_{q=1}^{Q} Y^{q}_{v_i v_j} \geq 1, \qquad \forall v_j \tag{6}$$

$$S_{qv} \geq r_q - M(1 - Y^{q}_{v_0 v}), \qquad \forall v, q \tag{7}$$

$$S_{qv} \geq T_v - M(1 - \sum_{v_j=1}^{V+1} Y^{q}_{v v_j}), \qquad \forall v, q \tag{8}$$

$$S_{q v_j} \geq C_{q v_i} - M(1 - Y^{q}_{v_i v_j}), \qquad \forall v_i, v_j; v_i \neq v_j; q \tag{9}$$

$$\sum_{b_j=1}^{B_v} X^{qv}_{b_0 b_j} = \sum_{v_i=0}^{V} Y^{q}_{v_i v}, \qquad \forall v, q \tag{10}$$

$$\sum_{b_j=1}^{B_v} X^{qv}_{b_j b_{B_v+1}} = \sum_{v_i=0}^{V} Y^{q}_{v_i v}, \qquad \forall v, q \tag{11}$$

$$\sum_{b_j=1}^{B_v+1} X^{qv}_{b b_j} - \sum_{b_j=0}^{B_v} X^{qv}_{b_j b} = 0, \qquad \forall b, v, q \tag{12}$$

$$\sum_{q=1}^{Q} \sum_{b_i=0}^{B_v} X^{qv}_{b_i b_j} = 1, \qquad \forall b_j, v \tag{13}$$

$$\sum_{b_i=0}^{B_v} \sum_{b_j=1}^{B_v+1} X^{qv}_{b_i b_j} \leq M \sum_{v_i=0}^{V} Y^{q}_{v_i v}, \qquad \forall v, q \tag{14}$$

$$D^{v}_{b_i} + p^{v}_{b_i} + t^{qv}_{b_i b_j} - D^{v}_{b_j} \leq M(1 - X^{qv}_{b_i b_j}), \qquad \forall b_i, b_j, v, q \tag{15}$$

$$S_{qv} + p^{v}_{b_j} + t^{qv}_{b_0 b_j} - D^{v}_{b_j} \leq M(1 - X^{qv}_{b_0 b_j}), \qquad \forall b_j, v, q \tag{16}$$

$$D^{v}_{b_j} - C_{qv} \leq M(1 - X^{qv}_{b_j b_{B_v+1}}), \qquad \forall b_j, v, q \tag{17}$$

$$C_{qv} - F_v \leq M(1 - \sum_{v_j=1}^{V+1} Y^{q}_{v v_j}), \qquad \forall v, q \tag{18}$$

$$D^{v}_{b_i} + p^{v}_{b_j} \leq D^{v}_{b_j}, \qquad \forall (b_i, b_j) \in \Phi_v; b_j \neq b_i; \forall v \tag{19}$$

$$D^{v}_{b_i} - D^{v}_{b_j} + p^{v}_{b_j} \leq M(1 - Z^{v}_{b_i b_j}), \qquad \forall b_i, b_j; b_i \neq b_j; \forall v \tag{20}$$

$$Z^{v}_{b_i b_j} + Z^{v}_{b_j b_i} = 1, \qquad \forall (b_i, b_j) \in \Psi_v; b_j \neq b_i; \forall v \tag{21}$$

$$\sum_{\theta=0}^{Q} \sum_{\kappa=0}^{B_v} X^{\theta v}_{\kappa b_j} - \sum_{\theta=0}^{Q} \sum_{\kappa=0}^{B_v} X^{\theta v}_{\kappa b_i} \leq M(Z^{v}_{b_i b_j} + Z^{v}_{b_j b_i}), \qquad \forall b_i, b_j; j \neq i; l_{b_i} < l_{b_j}; \forall v, q \tag{22}$$

$$P_{v_i} + l^{v}_{b_i} \leq P_{v_j} + l^{v}_{b_j} + M(1 - \alpha^{v_i v_j}_{b_i b_j}), \qquad \forall b_i, b_j, v_i, v_j; v_j \neq v_i \tag{23}$$

$$D^{v_i}_{b_i} - D^{v_j}_{b_j} + p^{v_j}_{b_j} \leq M(1 - \beta^{v_i v_j}_{b_i b_j}), \qquad \forall b_i, b_j, v_i, v_j; v_j \neq v_i \tag{24}$$

$$\beta^{v_i v_j}_{b_i b_j} + \beta^{v_j v_i}_{b_j b_i} + \alpha^{v_j v_i}_{b_j b_i} \geq \sum_{\kappa=0}^{Bv} X^{q_i v_i}_{\kappa b_j} + \sum_{\kappa=0}^{Bv} X^{q_j v_j}_{\kappa b_i} - 1, \qquad \forall b_i, b_j, v_i, v_j, q_i, q_j; v_j \neq v_i; q_i < q_j \tag{25}$$

$$X^{qv}_{b_i b_j}, Z^{v}_{b_i b_j}, Y^{q}_{v_i v_j}, \alpha^{v_i v_j}_{b_i b_j}, \beta^{v_i v_j}_{b_i b_j} \in \{0, 1\}, \tag{26}$$

$$C_{qv}, F_v, D^{v}_{b_j}, P_v, T_v \geq 0. \tag{27}$$

In the objective function (1), the first term $\sum_{v=1}^{V} W_v A_v$ represents the tardiness cost if the departure time of a vessel is greater than its due time. The second term $\sum_{v=1}^{V} R_v E_v$ represents the earliness income if the finishing time of a vessel is less than its due time. Note that in practice this reward for earliness may be zero. Constraints (2) calculate the earliness or tardiness of a vessel depending on the difference between its due time and its finishing time.

The constraints (3-6) represent the main conditions for QCAP. However, constraints (3) and (4) respectively select the first and the last ships for each quay crane. Constraints (5) guarantee that ships are processed in a well-defined sequence. Constraints (6) guarantee that each vessel be handled by at least one quay crane.

The constraints (7-9) determine the starting time of quay cranes. Constraints (7) force the starting time of the earliest vessel that is to be done by the q^{th} quay crane to be after the ready time of the q^{th} quay crane. Note that vessel v_0 is a dummy vessel from which the working sequence starts. Constraints (8) say that the starting time of the q^{th} quay crane on the vessel v is no earlier than the berthing time of vessel v if the q^{th} quay crane is assigned to serve this vessel. Constraints (9) ensure that the starting time of the q^{th} quay crane on vessel v_j is no earlier than the finishing time of its predecessor vessel v_i.

Constraints (10) ensure that if a quay crane is assigned to a vessel, then it will start its processing with one of the tasks on that vessel. Constraints (11) ensure that if a quay crane is assigned to a vessel,

then it will finish its processing with one of tasks on that vessel. Constraints (12) show a flow balance ensuring that tasks are performed in a well-defined sequence on every vessel. Constraints (13) ensure that every task on each vessel must be handled by exactly one quay crane. Constraints (14) ensure that tasks on a vessel are handled by a quay crane only if this quay crane is allocated to that vessel. Constraints (15) simultaneously determine the completion time for each task and eliminate sub-tours; sub-tours here are the looping on tasks which have already been done. To illustrate, let Task 1, Task 2, and Task 3, be carried out in this order. A sub-tour would be to do Task 1, Task 2, Task 3, and Task 2 again, for instance. Constraints (15) remove this possibility. Constraints (16) determine the quay crane starting time on vessel v and the completion time of the same quay crane is computed by constraints (17). Constraints (18) determine the finishing time of each vessel.

When required, constraints (19) force task b_i to be completed before the start of task b_j for all the task pairs $(b_i, b_j) \in \Phi$. Constraints (20) define $Z^v_{b_i b_j}$ such that $Z^v_{b_i b_j} = 1$ when the operation of task b_j on vessel v starts after the completion of task b_i on the same vessel. Constraints (21) ensure that the pair of tasks that are members of the set Ψ will not be handled simultaneously.

Constraints (22) remove the possibility of interference between quay cranes. Suppose that tasks b_i and b_j are performed simultaneously and $l_i < l_j$, then $Z^v_{b_i b_j} + Z^v_{b_j b_i} = 0$. Note that both quay cranes and tasks are ordered in an increasing order of their relative locations in the direction of increasing ship-bay number. Suppose that, for $q_1 < q_2$, quay crane q_1 performs tasks b_j and quay crane q_2 performs task b_i. Then, interference between quay cranes q_1 and q_2 results as in such case, $\sum_{\theta=1}^{q_1} \sum_{\kappa=0}^{Nv} X^{\theta v}_{\kappa j} - \sum_{\theta=1}^{q_1} \sum_{\kappa=0}^{Nv} X^{\theta v}_{\kappa i} = 1$. This violates constraints (22), since we have $Z^v_{b_i b_j} + Z^v_{b_j b_i} = 0$ as mentioned earlier.

Constraints (23) define $\alpha^{v_i v_j}_{b_i b_j}$ such that $\alpha^{v_i v_j}_{b_i b_j} = 0$ if the berthing position of vessel v_i plus the location of task b_i on that vessel is greater than the berthing position of vessel v_j plus the location of task b_j on that vessel. Figures 1 and 2 illustrate how the value of $\alpha^{v_i v_j}_{b_i b_j}$ is computed.

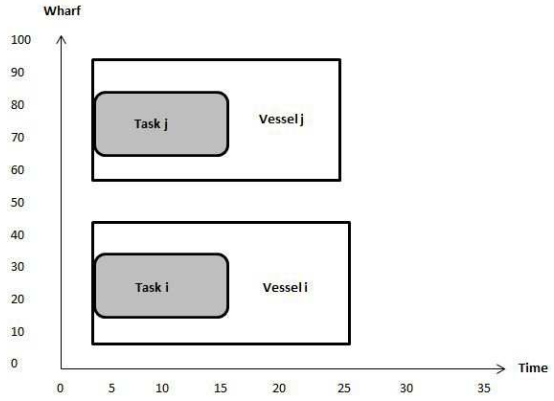

Figure 1: No location overlap between two vessels

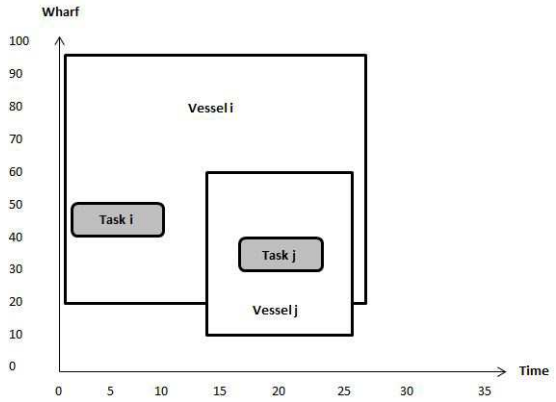

Figure 2: Location overlap between two vessels

The value of $\alpha^{v_i v_j}_{b_i b_j}$ in Figure 1 can be 1 because $P_{v_i} + l^v_{b_i} \leq P_{v_j} + l^v_{b_j}$. The value of $\alpha^{v_i v_j}_{b_i b_j}$ in Figure 2 equals 0, because $P_{v_i} + l^v_{b_i} > P_{v_j} + l^v_{b_j}$. This means there is overlap in the position between these two tasks on these two vessels.

Constraint (24) defines $\beta^{v_i v_j}_{b_i b_j}$ such that $\beta^{v_i v_j}_{b_i b_j} = 0$ if the finishing time of task b_i on vessel v_i plus the processing time of task b_j on vessel v_j is greater than the finishing time of task b_j on vessel v_j. Figures 3 and 4 illustrate how the value of $\beta^{v_i v_j}_{b_i b_j}$ is computed.

The value of $\beta^{v_i v_j}_{b_i b_j}$ in Figure 3 can be 1 because $D^{v_i}_{b_i} + p^{v_j}_{b_j} \leq D^{v_j}_{b_j}$, whereas the value of $\beta^{v_j v_i}_{b_j b_i}$ in the same figure equals 0 because $D^{v_j}_{b_j} + p^{v_i}_{b_i} > D^{v_i}_{b_i}$. The value of $\beta^{v_i v_j}_{b_i b_j}$ in Figure 4 equals 0 because $D^{v_i}_{b_i} + p^{v_j}_{b_j} > D^{v_j}_{b_j}$ and the value of $\beta^{v_j v_i}_{b_j b_i}$ in the same figure equals 0 because $D^{v_j}_{b_j} + p^{v_i}_{b_i} > D^{v_i}_{b_i}$. This means there is overlap in the time between these two tasks on these two vessels. Constraints (25) prevent the interference between quay cranes depending on the values of $\beta^{v_i v_j}_{b_i b_j}$ and $\alpha^{v_i v_j}_{b_i b_j}$, respectively.

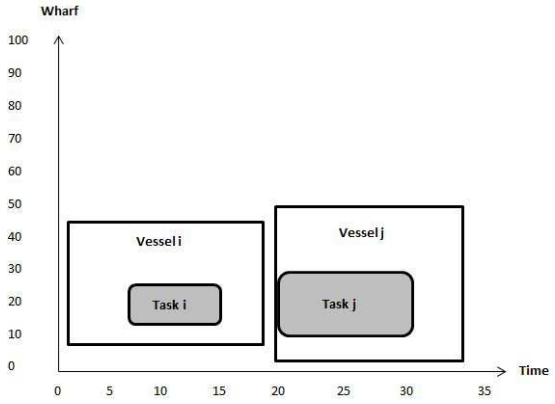

Figure 3: No location overlap between two vessels

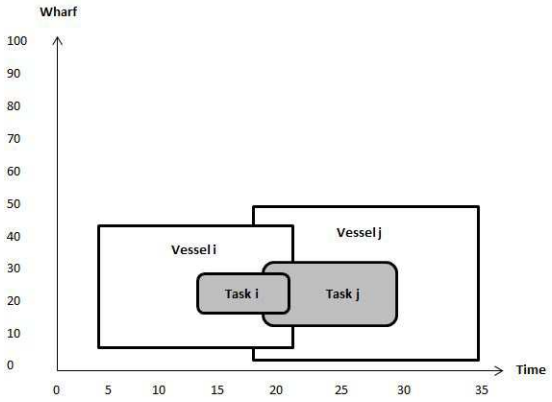

Figure 4: Location overlap between two vessels

3.7. Numerical examples

To illustrate the value of allowing quay cranes to move, in terms of the quality of plans, consider the situation in which two quay cranes are available to handle two vessels with data as given in Table 1. Each vessel has two tasks to be carried out. In the previous models where a fixed number of quay cranes is

Table 1: Input data of Example 1

Ready (crane)	0	2
Initial location (crane)	11	16
Arrival time	0	0
Processing time of tasks ,vessel 1	85	29
Processing time of tasks ,vessel 2	18	33
Location task, vessel 1	1	2
Location task, vessel 2	1	2
Expected departure time of vessel	85	33
Berthing position	10	15
Tardiness cost (per unit time)	5	1
Earliness income (per unit time)	1	1

allocated to every vessel for the duration of the processing period, the optimal working plan is described in Figure 5 with the objective value being equal to 171 (150 units of tardiness belong to the first vessel and 21 units of tardiness belong to the second vessel). Even though quay crane 2 finished its work on vessel 2 at 54 (2+18+1+33), it is not allowed to move away from vessel 2. This wastes the effective working time of this quay crane which results into a sub-optimal solution. In contrast, in our model, a variable number of quay cranes is used during the processing period, which allows the second quay crane to move to vessel 1 to perform other tasks as shown in Figure 6. As a result, vessel 1 processing is completed 87 time units earlier than in the previous plan with an objective value of 31 (10 units of tardiness are due to the first vessel and the other 21 units, to the second vessel); this is due to making the best use of both quay cranes. Note that we only allow the quay crane to move after it has finished its work on vessel 2.

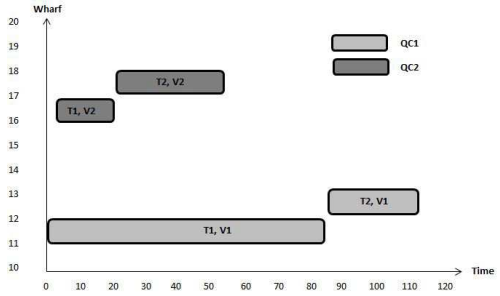

Figure 5: Previous model solution

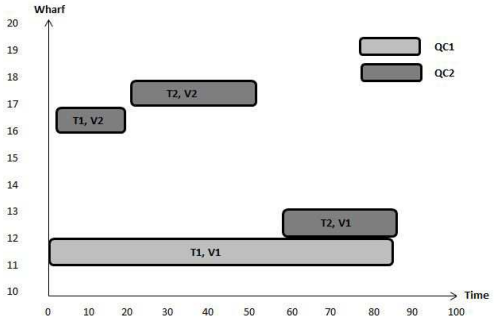

Figure 6: Suggested model solution

Our model also allows quay cranes to share tasks on the same vessel to which they are allocated. Consider the input data for two vessels arriving at a container terminal as given in Table 2. In Example

Table 2: Input data of Example 2

Ready (crane)	0	0
Initial location (crane)	22	24
Arrival time	0	0
Processing time of tasks,vessel 1	28	22
Processing time of tasks,vessel 2	39	34
Location task, vessel 1	1	2
Location task, vessel 2	1	2
Expected departure time of vessel	28	39
Berthing position	20	25
Tardiness cost (per unit time)	1	1
Earliness income (per unit time)	1	1

2, the objective value returned by previous models is 70 time units (33 units of tardiness are due to the first vessel and the other 37 units to the second vessel), (see Figure 7). Since we allow quay cranes to move between vessels if no interference constraints are violated, the solution of our model returns an objective value of only 35 time units (1 unit of tardiness due to the first vessel and 34 units due to the second vessel), as can be seen in Figure 8.

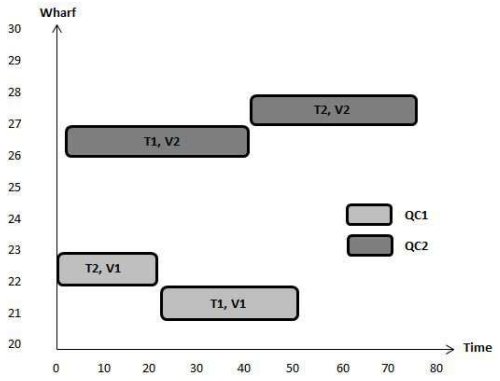

Figure 7: Previous model solution

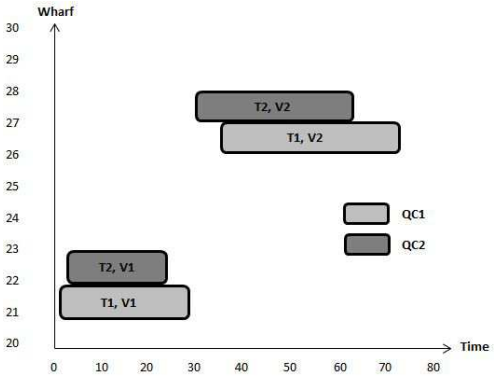

Figure 8: Suggested model solution

QCSP has been shown to be NP-hard by (Tavakkoli-Moghaddam et al., 2009; Lee et al., 2008; Garey and Johnson, 1979). Combining it with QCAP does not reduce the complexity of the joint problem since scheduling has still to be done. If anything, the increase in the size of the combined model due to the additional variables, some of which binary, and constraints, makes the problem no less difficult.

Therefore, large scale instances cannot be solved efficiently to optimality with exact approaches. Hence the need for an approximate approach. Here, we choose the Genetic Algorithm. GA is a well established meta-heuristic which has been shown to be very effective on combinatorial optimisation problems such as QCASP. Hence the choice. Given the novelty of the QCASP model we have put forward, an innovative implementation of GA is necessary to solve it. This implementation is explained below.

4. Application of the Genetic Algorithm to QCASP

Once an instance of the model is defined and the data is available, submitting it to CPLEX is straightforward. GA requires that we define a representation for the problem solutions as well as other algorithmic components necessary for its implementation as can be seen later. GA is an adaptive heuristic method based on evolution through natural selection ideas due to Darwin and others, (Holland, 1975). It is a population based approach, i.e. it searches for solutions by maintaining a population of solutions that are then updated from generation to generation using a number of possible genetic operators such as Crossover, Mutation, and Reproduction. Over successive generations, the population evolves towards an optimal solution. GA is particularly effective on difficult problems referred to as being NP-Hard. QCASP has been shown to be NP-hard, (Tavakkoli-Moghaddam et al., 2009; Lee et al., 2008; Garey and Johnson, 1979).

4.1. Solution representation: chromosome

GA starts with a randomly generated population of solutions. Each solution is called a chromosome and consists of a sequence of genes, here representing a sequence of holds (tasks) for all docked vessels. The value of a gene is randomly picked from the index set of all holds; it cannot, therefore, be duplicated, i.e. each gene is unique. Each chromosome consists of $v \times b$ genes, where v represents the number of vessels and b the number of tasks on each vessel. A simple chromosome for the case of two vessels, each with three tasks, is illustrated in Figure 9. Here, genes (1,2,3) represent the tasks on the first vessel and genes (4,5,6) represent those on the second vessel. Based on the sequence of tasks for all vessels

1	4	3	6	5	2

Figure 9: Chromosome representation

represented by the chromosome, a quay crane schedule can be constructed using the following steps that are the extension of the procedure proposed by (Lee et al., 2008) used for each vessel separately. In this paper, however, we assume that the berth allocation plan (berthing time and berthing position of each vessel arriving at the container terminal) and the initial position of each quay crane at the beginning of scheduling are known.

The quay crane scheduling procedure, (Lee et al., 2008):

Begin

Step 1: Based on the current position of each quay crane, determine which quay cranes can handle the first unassigned task in the chromosome without interference with other quay cranes. If only one quay crane is available, this task is assigned to this quay crane and it is deleted from the chromosome; the position and the completion time of the assigned quay crane are updated. The completion time of task i is also computed. If two quay cranes are available, go to Step 2.

Step 2: Compare the completion time of the two available quay cranes, and assign this task to the quay crane with earlier completion time. This task is then deleted from the chromosome, and both the position and the completion time of the assigned quay crane are updated. The completion time

of task i is computed. If their completion times are equal, go to Step 3.

Step 3: Compare the distances between this task and the two available quay cranes, respectively, and assign the task to the quay crane with the shorter distance. This task is deleted from the chromosome, and both the position and the completion time of the assigned quay crane are updated. The completion time of task i is also computed. If the distances are equal, go to Step 4.

Step 4: Assign this task to the quay crane with the smaller order number. Then, delete this task from the chromosome, and update both the position and the completion time of the assigned quay crane. Compute the completion time of task i.

Step 5: Steps 1–4 are repeated until all the tasks in the chromosome are assigned.

Stop

4.2. Solution validation

To validate chromosomes/solutions, three important situations must be considered. The first one is the precedence relationship between tasks. For instance, some of the bays of a given vessel may need to be unloaded and loaded. The discharging of containers from a bay must precede the loading of the bay. For this reason the generated chromosome should be checked to see if it satisfies this condition, i.e. constraints (19). The second situation is the non-simultaneity of some tasks, i.e. constraints (21) must also be satisfied. Finally, the interference between quay cranes is avoided by introducing non-interference constraints which consider both the potential interference of tasks on the same vessel (constraints (22)) and those on different vessels (constraints (25)). If either of constraints (19) or (21) or (22) or (25) are violated or both are violated, the generated chromosomes are discarded by adding a high penalty to their fitness values.

4.3. Evaluation of fitness

The objective of the QCASP is to minimise the tardiness of vessels. The completion time of each quay crane can be computed by summing up the processing time of all the tasks that have been performed by this quay crane plus the travel time which it takes to move from one hold to another. The tardiness of each vessel can be computed by subtracting the finishing time from the expected departure time. The finishing time represents the maximum processing time of the vessel required by the quay cranes assigned to it. The objective function used by the GA in Matlab is the same objective function as that of the mathematical model. Thus, the fitness value of a chromosome is calculated by Equation (28).

$$Fitness(chromosome) = \frac{1}{\sum_{v=1}^{V} W_v A_v - \sum_{v=1}^{V} R_v E_v} \tag{28}$$

4.4. Generating new populations

From the initial randomly generated population, subsequent generations of children, i.e. new populations, must be created. This is achieved by using genetic operators such as crossover, mutation and reproduction (copying of individuals unmodified into subsequent populations), (Salhi and Fraga, 2011; Salhi and Vazquez-Rodríguez, 2014; Vazquez-Rodríguez and Salhi, 2006b,a).

4.4.1. Selection process

The selection process picks chromosomes from the current population to be parents to new individuals (children/solutions) in the new population created using one of many genetic operators as listed above. To

give priority to the best chromosomes to pass their genes into the next generation, the fitness proportionate selection approach is implemented using a roulette wheel. High fitness individuals/solutions have high probability to be selected to contribute to the next population. In other words there is a bias toward their selection which means their genes are likely to be passed into the next generation.

4.4.2. Crossover operator

To produce a new chromosome (offspring) the 'Order Crossover' of (Cheng and Gen, 1997) is used. Order Crossover is a permutation-based crossover. It works as follows. A subsequence of consecutive alleles from parent 1 is selected and used to partially make the offspring; the remaining alleles to complete the creation of the offspring are chosen from parent 2 avoiding any repetitions. The same procedure is then applied starting from parent 2 to make the second offspring. The crossover operator always creates two offspring. Figures 10 and 11 illustrate the Order Crossover.

Parent1	7	12	5	3	6	1	10	8	11	2	4	9
Offspring1	**7**	**12**	**5**	**3**	8	2	6	4	11	9	10	1
Parent2	8	2	5	6	7	4	11	9	12	3	10	1

Figure 10: Offspring 1

Parent1	7	12	5	3	6	1	10	8	11	2	4	9
Offspring2	**8**	**2**	**5**	**6**	7	12	3	1	10	11	4	9
Parent2	8	2	5	6	7	4	11	9	12	3	10	1

Figure 11: Offspring 2

4.4.3. Mutation operator

To prevent the population getting trapped in a local optimum, the mutation operator is used to enable the GA search to escape and explore the search space globally by changing one or more genes. In this algorithm, two genes from the chromosome are randomly selected and then swapped with each other. Figure 12 illustrates the mutation operator.

8	4	**3**	6	5	2	7	**9**	1
8	4	**9**	6	5	2	7	**3**	1

Figure 12: Mutation operator

4.5. Stopping criteria

Two stopping criteria are used in the genetic algorithm: the maximum number of generations, and the number of generations without any improvement in the best solution found so far; this number is pre-set by the user.

5. Computational experiments

Twenty instances of QCASP with different numbers of vessels, tasks, and quay cranes have been solved using CPLEX and GA. The results are recorded in Tables 3 and 4. All instances have randomly generated processing time of tasks for each hold from the uniform distribution U(10,50). Berthing time and berthing position for each vessel are randomly generated in the interval [0, 100]. The initial location and ready (available) time for each quay crane are randomly generated in this interval. The weight of the tardiness

cost and earliness income is randomly generated in the interval [1, 10]. The expected departure time for each vessel is randomly generated considering the berthing time plus the total processing time of all tasks on it. The number of vessels, tasks and quay cranes are manually set up depending on the chosen problem size.

CPLEX solved problems 1 to 10. These are relatively small in size. They were solved in acceptable times, although 28, 68, 18 and 46 hours were required for problems 6 to 9, respectively. These times are hardly acceptable in the context of container terminal operations. The rest of the problems, 11 to 20, which are the realistic instances, could not be solved with CPLEX in acceptable times ($>$ 100 hours of CPU time). In some cases they could not be solved at all due to the limitations of the computing platform used. The runs of these instances were terminated after 3 hours.

GA, coded in Matlab, managed to solve all 20 instances. For the small size problems of Table 3, the GA parameters of population size, rate of crossover, rate of mutation, and the maximum number of generations are set to 150, 0.2, 0.1, and 500, respectively. In the case of the large size instances of Table 4, population size, crossover and mutation rates, and the maximum number of generations are set to 300, 0.25, 0.2, and 1000, respectively. These parameters have been set by experimentation and other experiences that can be found in the literature. Moreover, 20 trials were conducted to solve each instance using GA to mitigate the randomness of the algorithm.

All experiments have been performed on a PC with Intel Core i5 and 3.20 GHz CPU with 8 GB RAM running Windows 7 Operating System. The 20 problems and their corresponding results are presented in Tables 3 and 4, containing small problems 1 through 10 and problems 11 through to 20, respectively.

5.1. Results

On the small size problems of Table 3, the number of constraints, the number of decision variables, and the CPLEX computational time grow exponentially with the increase in the number of vessels, tasks and quay cranes. Note, however, that in column 13 of Table 3 showing the CPU time required by GA to solve these problems, this time does not change much with the increase in the problem size. It is also important to note that in most cases, GA obtains the optimal or near optimal solution.

Table 3: Computational results for small scale instances

No.	Problem Information			Problem Size			CPLEX		GA				Gap(%)
	Vessels	Tasks	Q.Cranes	Constraints	Dec.Vars	Int.Vars	Obj.Val	CPU(hh:mm:ss)	Best Obj.Val	Mean	St.Dev	CPU(s)	
1	2	3	5	538	270	243	70	00:00:05	70	70	0.0	15	0.0
2	4	2	5	1030	444	389	242	00:06:11	244	244	0.0	14	0.8
3	3	3	5	1141	474	431	405	00:40:54	405	412.5	8.3	13	0.0
4	2	5	5	1244	566	535	105	04:41:12	105	105	0.0	18	0.0
5	4	2	8	2152	636	560	156	00:34:34	156	156	0.0	14	0.0
6	4	3	4	1412	632	580	865	28:59:44	884	904.6	28.7	17	2.1
7	3	4	6	2472	807	756	1264	68:06:22	1268	1268	0.0	14	0.3
8	3	5	5	2776	1014	965	1035	33:56:21	1035	1038.5	20.9	15	0.0
9	3	5	8	6013	1392	1328	770	46:09:14	780	780	0.0	15	1.2
10	2	8	8	5972	1766	1720	320	06:05:31	320	320	0.0	18	0.0

CPLEX did not solve some of the large scale instances, shown in Table 4. GA, however, finds the optimal or near solutions for all the instances in reasonable CPU times (see column 13 of Table 4).

Table 4: Computational results for large scale instances

No.	Problem Information			Problem Size			CPLEX		GA			
	Vessels	Tasks	Q.Cranes	Constraints	Dec.Vars	Int.Vars	Obj.Val	CPU(hh:mm:ss)	Best Obj.Val	Mean	St.Dev	CPU(s)
11	5	10	8	67966	9675	9538	2275	03:00:00	667	670	2.5	93
12	5	8	12	94942	8235	8072	975	03:00:00	542	549.5	6.6	82
13	4	16	10	92060	11084	10954	1777	03:00:00	1030	1073.5	25.9	115
14	6	8	10	98336	9642	9466	3731	03:00:00	651	669.8	11.4	95
15	5	12	8	97426	13575	13428	2878	03:00:00	884	912.2	13.1	111
16	4	16	8	107096	16652	16520	4302	03:00:00	1112	1130.8	8.9	113
17	4	12	12	130400	12492	12348	2440	03:00:00	690	710.8	17.6	89
18	4	16	12	230784	21388	21228	not	responding	1026	1071.1	34.4	115
19	5	16	10	263700	26385	26200	not	responding	1379	1417.8	20.1	137
20	6	12	12	313236	22338	22116	not	responding	1045	1080.6	22.4	124

6. Conclusion

This paper describes QCASP, a mathematical formulation of the combined problem of Quay Crane Assignment and Quay Crane Scheduling. It is a mixed integer programming model which allows quay cranes to move between two holds of the same ship and between two holds on different vessels. The time it takes for these cranes to move between holds is taken into account in the optimisation process. Moreover, the model takes into account the fact that not all quay cranes start at the same time; some only become available after some delay due to other occupations. Therefore, the initial position for each quay crane is taken into account in order to compute the exact time of quay crane traveling. Precedence and simultaneity constraints are also taken into account. Interference between quay cranes is avoided by introducing non-interference constraints which consider both the potential interference of tasks on the same vessel and those on different vessels. The Branch-and-Cut algorithm as implemented in CPLEX 12.6 has been used to find the optimal solutions of relatively small instances of QCSAP. It cannot cope with larger instances of the problem which are of practical size. GA, however, coped well with all problems. It required almost the same CPU time for all problems of small size and CPU times of the same magnitude for the larger instances. Moreover, on most of the 10 instances that CPLEX managed to solve, GA also found the optimum. Overall the GA, however, coped well with all problems. It required almost the same CPU time for all problems of small size and CPU times of the same magnitude for the larger instances. This shows that GA, while substantially more efficient than B&C, is also quite robust on most instances considered as this average discrepancy shows.

The combined model presented here is obviously the way forward as it is more likely to provide better solutions than those found by solving seaside operations problems individually. It is also a substantial improvement on combined variants which do not allow for quay crane movement between vessels. However, there is scope for doing even better by combining Berth Allocation, Quay Crane Assignment and Quay Crane Scheduling into a single model. We are currently working on this problem and will report our findings in a future paper.

Acknowledgment

We would like to thank Tom Corkhill and Stephen Peck of the Port of Felixstowe for providing real instances of the problems considered, as well as practical insights into port operations. We also like to thank the Iraqi Ministry of Higher Education and Scientific Research for sponsoring this work.

References

Alsoufi, G., Yang, X., and Salhi, A. (2015). A combinatorial benders' cuts approach to the seaside operations problem in container ports. *Presented in the 11th metahueristics international conference, Agadir, Morocco, june 7-10.*

Alsoufi, G., Yang, X., and Salhi, A. (2016). Robust berth allocation using a hybrid approach combining branch-and-cut and the genetic algorithm. In *International Workshop on Hybrid Metaheuristics*, pages 187–201. Springer.

Bierwirth, C. and Meisel, F. (2009). A fast heuristic for quay crane scheduling with interference constraints. *Journal of Scheduling*, 12(4):345–360.

Bierwirth, C. and Meisel, F. (2010). A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 202(3):615–627.

Cheng, R. and Gen, M. (1997). Parallel machine scheduling problems using memetic algorithms. *Computers & Industrial Engineering*, 33(3-4):761–764.

Cheong, C. Y., Habibullah, M. S., Goh, R. S. M., and Fu, X. (2010). Multi-objective optimization of large scale berth allocation and quay crane assignment problems. In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 669–676. IEEE.

Chung, S. and Choy, K. L. (2012). A modified genetic algorithm for quay crane scheduling operations. *Expert Systems with Applications*, 39(4):4213–4221.

Daganzo, C. F. (1989). The crane scheduling problem. *Transportation Research Part B: Methodological*, 23(3):159–175.

Diabat, A. and Theodorou, E. (2014). An integrated quay crane assignment and scheduling problem. *Computers & Industrial Engineering*, 73:115–123.

Feo, T. A. and Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133.

Fu, Y.-M., Diabat, A., and Tsai, I.-T. (2014). A multi-vessel quay crane assignment and scheduling problem: Formulation and heuristic solution approach. *Expert Systems with Applications*, 41(15):6959–6965.

Garey, M. R. and Johnson, D. S. (1979). Computers and intractability: A guide to the theory of np-completeness.

Guan, Y. and Cheung, R. K. (2004). The berth allocation problem: models and solution methods. *OR Spectrum*, 26(1):75–92.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* (2nd ed. in 1992). Cambridge: MIT Press.

Kaveshgar, N., Huynh, N., and Rahimian, S. K. (2012). An efficient genetic algorithm for solving the quay crane scheduling problem. *Expert Systems with Applications*, 39(18):13108–13117.

Kenan, N. and Diabat, A. (2015). A branch-and-price algorithm to solve a quay crane scheduling problem. *Procedia Computer Science*, 61:527–532.

Kim, K. H. and Park, Y.-M. (2004). A crane scheduling method for port container terminals. *European Journal of operational research*, 156(3):752–768.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press.

Lee, D.-H., Wang, H. Q., and Miao, L. (2008). Quay crane scheduling with non-interference constraints in port container terminals. *Transportation Research Part E: Logistics and Transportation Review*, 44(1):124–135.

Legato, P., Gullì, D., and Trunfio, R. (2008). The quay crane deployment problem at a maritime container terminal. In *Submitted to the 22th European Conference on Modelling and Simulation.*

Meisel, F. and Bierwirth, C. (2009). Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196–209.

Moccia, L., Cordeau, J.-F., Gaudioso, M., and Laporte, G. (2006). A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal. *Naval Research Logistics (NRL)*, 53(1):45–59.

Nguyen, S., Zhang, M., Johnston, M., and Chen Tan, K. (2013). Hybrid evolutionary computation methods for quay crane scheduling problems. *Computers & Operations Research*, 40(8):2083–2093.

Peterkofsky, R. I. and Daganzo, C. F. (1990). A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B: Methodological*, 24(3):159–172.

Salhi, A. and Fraga, E. S. (2011). Nature-inspired optimisation approaches and the new plant propagation algorithm. In *The ICeMATH2011*, pages K2–1–K2–8.

Salhi, A. and Vazquez-Rodríguez, J. A. (2014). Tailoring hyper-heuristics to specific instances of a scheduling problem using affinity and competence functions. *Memetic Computing*, 6(2):77–84.

Sammarra, M., Cordeau, J.-F., Laporte, G., and Monaco, M. F. (2007). A tabu search heuristic for the quay crane scheduling problem. *Journal of Scheduling*, 10(4-5):327–336.

Steenken, D., Voß, S., and Stahlbock, R. (2004). Container terminal operation and operations research-a classification and literature review. *OR spectrum*, 26(1):3–49.

Tavakkoli-Moghaddam, R., Makui, A., Salahi, S., Bazzazi, M., and Taheri, F. (2009). An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. *Computers & Industrial Engineering*, 56(1):241–248.

Theodorou, E. and Diabat, A. (2015). A joint quay crane assignment and scheduling problem: formulation, solution algorithm and computational results. *Optimization Letters*, 9(4):799–817.

Unsal, O. and Oguz, C. (2013). Constraint programming approach to quay crane scheduling problem. *Transportation Research Part E: Logistics and Transportation Review*, 59:108–122.

Vazquez-Rodríguez, J. A. and Salhi, A. (2006a). Hybrid evolutionary methods for the solution of complex scheduling problems. In *Advances in Artificial Intelligence*, pages 17–28. Springer.

Vazquez-Rodríguez, J. A. and Salhi, A. (2006b). A synergy exploiting evolutionary approach to complex scheduling problems. *Computer Aided Methods in Optimal Design and Operations, Series on Computers and Operations Research, World Scientific Publishing Co. Pvt. Ltd*, pages 59–68.

Yang, C., Wang, X., and Li, Z. (2012). An optimization approach for coupling problem of berth allocation and quay crane assignment in container terminal. *Computers & Industrial Engineering*, 63(1):243–253.